CLAIMS:

1. A method of providing redundant logic paths on a programmable logic device to mitigate radiation induced errors, the method comprising the steps of:

   (a) providing a plurality of instances of a logic function each having the same at least one logic output signal; and

   (b) sending each same at least one output signal through a chip output comprising a tri-state output buffer having an enable control operable to enable and disable the associated chip output, the enable control being controlled by a voting circuit which receives each same at least one output signal and is operable to produce an enable control output based upon the number of inputs having the same value.

2. The method as set forth in claim 1, further comprising the step of wiring the logic signals together after the logic signals have left the device.

3. The method as set forth in claim 1, the enable control output of the voting circuit being based upon whether the input value received from the associated logic instance is a minority of all the same at least one logic outputs.

4. An electronic circuit for maintaining the protective redundancy of logic paths until the logic paths have left a device upon which the logic paths were traveling, the circuit comprising redundant logic paths, the redundant logic paths comprising:

a plurality of identical instances of logic, each producing at least one data output signal; and

at least one chip output associated with each logic instance and receiving the associated at least one output, the chip output comprising a tri-state output buffer having an enable control, the enable control being controlled by a voting circuit which receives each at least one data output signal from the logic instances and is operable to produce an enable control output based upon the number of inputs having the same value.

5. The electronic circuit of claim 4, the enable control output of the voting circuit being based upon whether the associated at lest one data output signal is a minority of all of the at least one data output signals.

6. The electronic circuit of claim 4, the logic signals being wired together after leaving the device.

7. A method of combining a plurality of logic signals intended to have the same value into a single signal traveling on a data line, each logic signal having a logic value and traveling on a logic path, the method comprising the steps of:

(a) determining whether each logic signal has a desired value;

(b) enabling each logic path on which the logic signal has the desired value to allow the logic signal to move to the data line;

(c) disabling each logic path on which the logic signal does not have the desired value from moving to the data line; and

(d) connecting all of the logic paths to form the single data line.

8. A method of protecting a system from the effects of radiation on a logic design, the logic design being configured in a configurable device and providing input via at least one data line to the system, the method comprising the steps of:

(a) establishing redundant instances of the logic design, the logic design and redundant instances thereof each producing at least one logic output signal having a logic value and traveling on a logic path;

(b) determining whether each logic output signal has a desired logic value;

(c) enabling each logic path on which the logic signal has the desired value to

allow the logic signal to move to the data line;

(d) disabling each logic path on which the logic signal does not have the desired value from moving to the data line; and

(e) connecting all of the logic paths to form the single data line.

9. The method as set forth in claim 8, steps (c) and (d) being performed by a buffer connected to each logic path, the buffer having an enable control that regulates whether the buffer allows the logic signal to move to the data line.

10. The method as set forth in claim 8, step (b) being performed by at least one majority voting circuit.

11. A method ensuring correct output from a first logic design, the method comprising the steps of:

(a) configuring the first logic design and a redundant instance thereof in a first configurable device, the first logic design and redundant instance thereof each producing at least one logic output on a first logic path connected to a data line;

(b) configuring a second logic design and a redundant instance thereof in a second configurable device, the second logic design being identical to the first logic design, the second logic design and the redundant instance thereof each

producing at least one logic output on a second logic path connected to the data line;

(c) synchronizing the first and second devices;

(d) comparing the logic output of the first logic design to the logic output of the redundant instance thereof to produce a first comparison result;

(e) reconfiguring the first logic function and the redundant instance thereof if the first comparison result is a predetermined value;

(f) re-synchronizing the first and second devices if the first logic design has been reconfigured;

(g) comparing the output of the second logic design to the output of the redundant instance thereof to produce a second comparison result;

(h) reconfiguring the second logic design and the redundant instance thereof if the comparison result is a predetermined value; and

(i) re-synchronizing the first and second devices if the second logic design has been reconfigured.

12. A dual device circuit operable to ensure correct output from a logic design, the dual device circuit comprising:

a first logic design and a redundant instance thereof each configured in a first device, the first logic design and the redundant instance thereof each being operable to produce a logic output;

a second logic design and a redundant instance thereof each configured in a second device, with the second logic design being identical to the first logic design, the second logic design and the redundant instance thereof each being operable to produce a logic output;

a synchronizer operable to synchronize the first and second devices;

a first comparator operable to compare the output of the first logic design to the output of the redundant instance thereof and to produce a first comparison result;

a second comparator operable to compare the output of the second logic design to the output of the redundant instance thereof and to produce a second comparison result;

a first reconfiguration circuit operable to cause the first logic design and the redundant instance thereof to reconfigure in the first device if the first comparison result is a predetermined value, the first reconfiguration circuit being further

operable to cause the synchronizer to synchronize the first and second devices if the first logic design has been reconfigured; and

a second reconfiguration circuit operable to cause the second logic design and the redundant instance thereof to reconfigure in the second device if the second comparison result is a predetermined value, the second reconfiguration circuit being further operable to cause the synchronizer to synchronize the first and second devices if the second logic design has been reconfigured.

13. The dual device circuit of claim 12, further comprising a watchdog timer operable to cause the synchronizer to synchronize the first and second devices after a predetermined number of clock cycles.

14. A method of ensuring correct output from a looped logic design, the logic design being configured in at least three redundant instances thereof, the method comprising the steps of:

(a) configuring the logic design and redundant instances thereof, the logic design and redundant instances thereof each producing at least one logic output and each having at least one feedback

path from the logic output to a logic input; and

(b) mitigating the logic outputs using at least one voting circuit operable to receive the logic outputs of the logic design and redundant instances thereof and to produce a consensus output based upon the number of logic outputs having a certain value wherein the voting circuit output is provided to the at least one feedback path of each logic instance.

15. The method as set forth in step 14, wherein each redundant logic instance is configured in the same device.

16. The method of claim 14, further comprising providing a voting circuit for each logic instance.

17. A method of ensuring correct output from a logic design, the logic design being configured in a single configurable device, the method comprising the steps of:

(a) configuring two redundant instances of the logic design in the same device and three redundant instances in each of a plurality of configurable devices, with the logic design and redundant instances thereof in each device each producing at least one logic output;

(b) using at least one mitigation device to mitigate the logic outputs.

18. The method as set forth in claim 17, the total number of configurable devices configured with the logic design and redundant instances thereof being three.

19. The method as set forth in claim 17, the mitigation device being a microprocessor.

20. The method as set forth in claim 17, the mitigation device being an application specific integrated circuit.

21. The Method as set forth in claim 17, the logic design being a portion of a larger complete logic design.

22. The method as set forth in claim 17, wherein each logic instance contains a logic loop having a node corresponding to the logic loop and at least one voting circuit determines the majority output of the logic loop nodes.

23. A method of maintaining configuration data in a configurable device, the method comprising the steps of:

(a) performing a first readback of a portion of the configuration data and generating a first checksum value indicative thereof;

(b) performing a second readback of the same portion of the configuration data and generating a second checksum value indicative thereof; and

(c) comparing the first checksum value with the second checksum value and, if they are not identical, indicating that remedial action should be taken.

24. The method as set forth in claim 23, the method further comprising the step of initiating a reconfiguration of the configurable device when remedial action is indicated.

25. The method as set forth in claim 23, the comparison being performed by a microcontroller.

26. The method as set forth in claim 23, the comparison being performed by a second configurable device, the first and second configurable devices being initially configured with identical configuration data.

27. A method of maintaining correct configuration data in a configurable device, the method comprising the steps of:

(a) performing a readback of a portion of the configuration data and generating a checksum value indicative thereof;

(b) comparing the first checksum value with a predetermined master checksum value,

the master checksum value being known to be correct; and

(c) indicating, if the comparison reveals that the checksum values are not identical, that remedial action should be taken.

28. The method as set forth in claim 27, the method further comprising the step of initiating a reconfiguration of the configurable device when remedial action is indicated.

29. The method as set forth in claim 27, the comparison being performed by a microcontroller.

30. A method of correcting an error in configuration data in a configurable device, the error being confined to a portion of the configuration data, the method comprising the steps of:

(a) identifying the portion of the configurable data containing the error; and

(b) reconfiguring only the portion of the configuration data containing the error.

31. The method as set forth in claim 30, the portion of the configuration data containing the error being a single data frame.

32. The method as set forth in claim 30, step (a) being accomplished using limited readback and comparison with a checksum value.

33. A method of correcting an expected error in configuration data in a configurable device, the configuration data being subject to periodic errors, the method comprising the steps of:

(a) determining the upset frequency with which an error is likely to occur; and

(b) reconfiguring the configuration data with a scrub frequency based upon the determined error frequency and without determining whether an error has actually occurred.

34. The method as set forth in claim 33, the configuration data being exposed to radiation likely to cause the expected error, the error frequency being determined based upon length of exposure of the configuration data to the radiation and the expected strength of the radiation.

35. The method as set forth in claim 33, the scrub frequency being one order of magnitude greater than the error frequency.

36. A method of correcting radiation induced errors in a RAM comprising:

providing a plurality of instances of a RAM each having dual ports capable of input and output;

providing one redundant clock signal to each RAM instance;

using a fist port of each RAM instance for refreshing data; and

using a second port of each RAM instance for user logic access.

37. The method of claim 36, further comprising synchronously reading each RAM instance from the first port.

38. The method of claim 37, further comprising supplying the majority value from reading the redundant RAM instances to the input of the first port of each RAM instance.

39. A method of correcting radiation induced errors in a clock delay locked loop (DLL) comprising:

providing at least three of instances of a DLL;

providing a one bit counter for each DLL instance, each one bit counter being clocked by the associated DLL;

determining whether the phases of all one bit counter outputs are the same; and

resetting a DLL if the associated one bit counter is not in phase with a majority of the one bit counters.

40. The method of claim 39, the step of determining being performed by a voting circuit associated with each one bit counter.

41. The method of claim 40, the voting circuit analyzing whether its associated one bit counter output is a minority of the output values of the other one bit counters.

42. The method of claim 39, further comprising providing a master clock signal redundantly to each DLL instance.

43. The method of claim 39, further comprising enabling each one bit counter only when each DLL instance has signaled it is phase locked.

44. The method of claim 43, further comprising resetting each one bit counter when a DLL is reset.

45. A method of ensuring an FPGA device output is not erroneously asserted comprising:

configuring at least two instances of the same logic function each having a device output; and

tying the device outputs of each logic instance together as inputs to a wire AND configuration such that the wire AND output will only be high when each logic instance output is high.

46. A method of ensuring an FPGA device output is not erroneously asserted comprising:

configuring at least two instances of the same logic function each having a device output; and

providing the device outputs of each logic instance to a logic gate such that the logic gate output will only be active when each logic instance provides an active output signal.

47. The method of Claim 46 wherein the logic gate is an OR gate and the active state is logic low.

48. The method of Claim 46 wherein the logic gate is a NAND gate, the active output signals of the logic instances are high, and the active output signal of the NAND gate is logic low.